AF/IFW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No.: 09/660,005
Filed: September 12, 2000
Inventor:
Thomas E. Saulpaugh
Gregory L. Slaughter
Michael J. Duigou

Title: PRE-GENERATED MESSAGE ENDPOINTS

Examiner: Phan, Tam T.
Group/Art Unit: 2144
Atty. Dkt. No: 5181-66200

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date indicated below.

Robert C. Kowert
Name of Registered Representative

Signature — September 1, 2005 (Date)

PRE-APPEAL BRIEF REQUEST FOR REVIEW

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Applicants request review of the final rejection in the above-identified application. No amendments are being filed with this request. This request is being filed with a notice of appeal. The review is requested for the reason(s) stated below.

Claims 1-28 remain pending in the application. Reconsideration of the present application is earnestly requested in light of the following remarks. Please note that for brevity, only the primary arguments directed to the independent claims are presented, and that additional arguments, e.g., directed to the subject matter of the dependent claims, will be presented if and when the case proceeds to Appeal.

Claims 1-7, 14-20 and 27 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Juster et al. (U.S. Patent 6,202,089) (hereinafter "Juster") in view of Stern et al. (U.S. Patent 5,935,249) (hereinafter "Stern"). Claims 8-13, 21-26 and 28 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Hind, et al. (U.S. Patent 6,585,778) (hereinafter "Hind") in view of Lee, et al. (U.S. Patent 6,336,137) (hereinafter "Lee"). **Applicants submit that the Examiner has failed to support a *prima facie* rejection of each of the independent claims.** Applicants note the following clear errors in the Examiner's rejection.

Regarding claims 1, 14 and 27, contrary to the Examiner's assertion, Juster in view of Stern fails to teach or suggest receiving an address for a service within the distributed computing environment; *linking* the address to a *pre-generated message interface* for accessing the service, wherein the message

interface comprises computer-executable code built into the device, and wherein *the linking creates* a message endpoint for the device to send messages to the service at the address in order to access the service; and using the message endpoint to send messages to the address to access the service.

Juster teaches a method for configuring, identifying, and using a plurality of remote procedure call (RPC) endpoints within a single server process. Specifically, Juster teaches a server process that establishes a plurality of RPC endpoints and a single, separate RPC endpoint for responding to address queries from clients. According to Juster, a client first places a remote procedure call to the address request RPC endpoint of the server process that returns the address of another RPC endpoint providing a service desired by the client (Juster, Abstract, column 2, lines 3-26).

The Examiner cites the Abstract, Figures 2A, 2B, and column 2, lines 3-26 and column 7-lines 10-31 of Juster. However, none of the cited portions of Juster disclose *linking a received address to a pre-generated message interface* for accessing the service. Instead, Juster teaches that a client queries the server via the server's address request RPC endpoint to obtain the address of another server RPC endpoint that provides a desired service (Juster, column 2, lines 18-23, column 4, lines 55-52, and column 7, lines 10-31). Once the client obtains the address, Juster states only that the client performs remote procedure calls on the server RPC endpoint using the received address (Juster, Figure 5, step 525, column 2, lines 23-25, and column 4, lines 62-64). Juster does not mention anything about *linking* the address received from the server *to a pre-generated message interface* for accessing the service. **In fact, Juster is completely silent regarding how a client performs remote procedure calls.** Juster is not concerned with how a client performs remote procedure calls. Instead, Juster deals only with configuring multiple *server* RPC endpoints for a single server process.

The Examiner admits that Juster fails to teach a pre-generated message interface comprising computer-executable code built into the device and relies upon Stern, citing the Title, Figures 4, 6, 7, and 9, as well as column 4, lines 1-14 and column 8, lines 27-59 of Stern. However, none of the cited portions of Stern disclose a *pre-generated message interface comprising computer-executable code built into the device.* Please see Applicants' Response to Final Action for a detailed description of Stern's teachings.

In response to the above argument, the Examiner states, "Juster is relied upon to disclose the linking feature and Stern disclosed the pre-generated message interface feature". Specifically, the Examiner argues, "the process of establishing an RPC endpoint portal for accessing a service reads on linking an address to an interface for accessing a service. **However, contrary to the Examiner's assertion, Juster does not mention or suggest anything about a client establishing an RPC endpoint *portal.*** Instead, as noted above, Juster merely states that a client places an remote procedure call to a server endpoint. Juster does not provide any specific teachings regarding how the client actually performs the remote procedure call.

Traditional remote procedure call (RPC) systems, such as those mentioned by both Juster and Stern, utilize one of two different client-side RPC mechanisms. In some traditional RPC systems, a client wishing to communicate with a remote server locates and downloads a stub object that is configured to communicate with the remote server object (or process). The client then executes methods of the stub object that communicates the necessary method parameters and return values between the client and the remote object. In other traditional RPC systems, the client repeatedly calls a standard RPC library routine, passing in the remote RPC endpoint address and any necessary parameters. The client calls the same routine to execute RPC calls on different remote server objects, passing in the relevant address each time. **Conventional RPC mechanisms, such as described in Juster and Stern, do not *link an address to a pre-generated message interface* for accessing a service, wherein the message interface comprises computer-executable code *built in to the device.***

The Examiner further responds to applicants arguments by referring to Stern's use of a "Java Enabled Network Interface Device having executable instruction built into the embedded interface of the device" and additionally cites column 2, lines 59-67 of Stern. Column 2, lines 59-67 of Stern describes how Stern's secure language processor, which is implemented with a Java language interpreter, provides control over embedded non-volatile memory for storing objects of value and a restricted interface and how "this mechanism allows a network interface device to represent and proxy for services available on a network, even when the local device is disconnected from the network." **However, neither this passage nor any other passage of Stern mentions a pre-generated message interface.** As described previously, the other portions of Stern cited by the Examiner also fail to mention any pre-generated message interface comprising computer-executable code built into the device. **In fact, Stern is completely silent regarding a pre-generated message interface for accessing a service including computer-executable code built into the device.** Without some clear teaching by Stern regarding a pre-generated message interface for access a service, the Examiner's argument amounts to nothing more than improper speculation regarding the workings of Stern's system. For a more detailed argument regarding Stern's teaching, please refer to Applicants' Response to Final Office Action.

Furthermore, Juster in view of Stern further fails to teach linking an address to a pre-generated message interface for accessing the service, wherein said linking creates a message endpoint for the device to send messages to the service at the address in order to access the service, contrary to the Examiner's assertion. As noted above, both Juster and Stern merely mention the use of remote procedure calls without describing any particular method of performing such a call. Juster states only, "the client then places a remote procedure call on the desired endpoint of the server process" (Juster, column 2, lines 23-26 and column 7, lines 39-46). Stern only mentioned the use of remote procedure calls in his background section to specifically point out deficiencies of using remote procedure calls (Stern, column 2, lines 18-34). Furthermore, as noted above, traditional RPC mechanisms, such as those mentioned by both Juster and Stern, do not utilize *creating* a message endpoint to send *multiple* messages *by linking* an address to a pre-generated message interface, as recited in Applicants' claims. Instead, as noted above traditional RPC mechanisms use stub objects downloaded from the server, and hence do not include code built in to the client device. Alternatively, traditional RPC mechanisms pass the address into an RPC library or API function with each call, and thus do not result in a single created message endpoint for sending *multiple* messages to the address.

The combination of Juster and Stern suggested by the Examiner would only result in a system including a server process that establishes multiple RPC endpoints, including an address query endpoint, and that also includes a Java Enabled Network Interface Device as taught by Stern. The rejection is clearly unsupported by the cited art since neither Juster nor Stern, alone or in combination, teach or suggest anything regarding linking an address to a pre-generated message interface for accessing the service, wherein the message interface comprises computer-executable code built in to the device, and wherein the linking creates a message endpoint for the device to send messages to the service at the address in order to access the service,.

In regard to claims 8, 21 and 28, contrary to the Examiner's assertion, Hind in view of Lee fails to teach or suggest a method for accessing services, comprising: receiving a schema defining messages for accessing the service and generating message endpoint code according to said schema.

Hind teaches a system for data policy enforcement using style sheet processing wherein a Document Type Definition (DTD) including stored policy enforcement objects is applied to an *input document* representing a response to a user request. As described in detail in Applicant's Response to Final Action, none of the Examiner's cited passages describe receiving a schema defining

messages for accessing a service. In contrast, Hind teaches a system that enforces data policy on retrieved documents (input documents resulting from user requests). **Hind does not teach anything regarding schemas that define messages for accessing services.** Instead Hind deals with data documents that result from user requests.

Hind in view of Lee also fails, contrary to the Examiner's contention, to teach generating message endpoint code according to the schema. Hind teaches the loading and executing of data policy objects that are referenced in a DTD defining data policies to be enforced on an input document. Such data policy objects are **not** message endpoint code, but instead are *style sheets* that translate, transform, or tailor the information of *input documents* before they are delivered to particular devices (Hind, column 7, lines 29-49, column 8, lines 38-57). Style sheet processing and data transformation cannot be considered to teach or suggest the generation of message endpoint code according to a schema defining messages for accessing a service. Hind's DTDs do not define messages for accessing a service and the transformation of Hind's input documents do not result in the generation of message endpoint code.

In response to the above arguments, the Examiner states, "Hind disclosed the use of template representation for accessing [a] service in a wireless system and the generating of [an] endpoint to establish communication path between the wireless device and the wireless service provider." **However, this statement by the Examiner is not supported in Hind and fails to rebut any of Applicants' arguments regarding the portions of Hind cited by the Examiner. Additionally, the Examiner argument fails to point out any portion of Hind that teaches or suggests generating message endpoint code according to a received schema defining messages for accessing a service. As noted above, Hind teaches uses DTDs to transform input documents to output documents.**

The Examiner admits that Hind does not teach linking said message endpoint code into executable operating code for the device and loading the message endpoint code and operating code onto the device and contends that Lee teaches such functionality. However, Lee also fails to teach or suggest linking generated message endpoint code into executable operating code for the device and loading the message endpoint code and operating code onto the device. Instead, Lee discloses a method for transferring data via a network between a server and clients or browsers that are spatially distributed wherein a server parses client requests to determine both the language of the request and the information requested. The server also associates various markup languages with different virtual directories and clients that want to request a particular markup language can route their requests to the appropriate directory. (Lee, Abstract, column 4, lines 13-34).

The Examiner cites two passages of Lee. The first passage describes how a client may be an HTTP client and also a gateway server to wireless browser clients that request pages from the client via WAP protocol that the gateway server transforms into HTTP/WML requests that are submitted to the main server. Such a gateway server also transforms the responses back into WAP/WML responses to be returned to the wireless clients (Lee, column 9, lines 21-45). The second passage cited by the Examiner describes a web engine that interprets a WML template containing embedded tags regarding what data the engine should retrieve from an associated database. The web engine then generates new WML code segments with the requested data and replaces the tags in the original WML template with the new code and sends the completed WML file to a WML browser (Lee, column 12, lines 22-38).

The Examiner's cited passages have no relevance to linking generated message endpoint code into executable operating code for a device and loading the massage endpoint code and operating code onto the device. In contrast, Lee is teaching the use of specialized SWE tags for building custom data responses in a client requested mark up language. Lee fails to teach anything regarding any type of linking or loading of message endpoint code.

Hind and Lee, separately and in combination, fail to teach or suggest receiving a schema defining messages for accessing a service; generating message endpoint code according to the schema; and linking the message endpoint code into executable operating code for the device and loading the message endpoint code and operating code onto the device.

The Examiner's rejection of many of the dependent claims is additionally erroneous. For example, the cited art is clearly insufficient to support the rejection of claims 2-7 and 15-20, as discussed in detail in Applicants' previous response on pp. 14-21. The cited is further insufficient to support the rejection of claims 9,10, 22 and 23, as discussed on pp. 25-27 of Applicants' previous response.

In light of the foregoing remarks, Applicant submits the application is in condition for allowance, and notice to that effect is respectfully requested. If any extension of time (under 37 C.F.R. § 1.136) is necessary to prevent the above referenced application from becoming abandoned, Applicants hereby petition for such an extension. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert & Goetzel PC Deposit Account No. 501505/5181-66200/RCK.

Also enclosed herewith are the following items:

☒ Return Receipt Postcard
☒ Notice of Appeal

Respectfully submitted,

Robert C. Kowert
Reg. No. 39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: September 1, 2005